

A

THE COMMISSIONER OF PATENTS AND TRADEMARKS  
Washington, D.C. 20231

04/22/98

Transmitted herewith for filing is the patent application of  
Inventor(s): **Subhash C. Roy, Paul Hembrook, Eugene L. Parrella and  
Richard Mariano**

For: **REAL TIME DEBUGGER INTERFACE FOR EMBEDDED SYSTEMS**

Enclosed are:

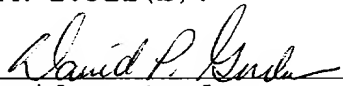
- ☒ 2 sheets of drawings.
- ☒ Assignment Recordation Sheet
- ☒ Assignment of the Invention to TranSwitch Corporation
- ☒ 2 verified statements to establish small entity status under 37 CFR 1.9 and 37 CFR 1.27.
- ☒ A Declaration and Power of Attorney document.
- ☐ Preliminary Amendment and Remarks

The filing fee has been calculated as shown below:

			SMALL ENTITY		OR	LARGE ENTITY	
FOR	NO. FILED	NO. EXTRA	RATE	FEE		RATE	FEE
BASIC FEE				\$ 395			\$ 790
TOTAL CLAIMS	25-20	5	X 11	55		X 22	198
INDEP CLAIMS	3- 3	0	X 41			X 82	
MULT. DEPENDENT CLAIMS PRESENTED			+135			+270	
			TOTAL \$450			TOTAL \$988	

- ☐ Please charge my Deposit Account No. \_\_\_\_\_ in the amount of \$\_\_\_\_\_. A duplicate copy of this sheet is enclosed.
- ☒ A check in the amount of \$ 490.00 to cover the filing/assignment recordation fee is enclosed.
- ☒ The Commissioner is hereby authorized to charge payment of the following fees associated with this communication or credit any overpayment to Deposit Account No. 07-1732. A duplicate copy of this sheet is enclosed.
- ☒ Any additional filing fees required under 37 CFR 1.16.
- ☐ Any patent application processing fees under 37 CFR 1.17.
- ☐ The Commissioner is hereby authorized to charge payment of the following fees during the pendency of this application or credit any overpayment to Deposit Account No. \_\_\_\_\_. A duplicate copy of this sheet is enclosed.
- ☐ Any patent application processing fees under 37 CFR 1.17.
- ☐ The issue fee set in 37 CFR 1.18 at or before mailing of the Notice of Allowance, pursuant to 37 CFR 1.311(b).

65 Woods End Road  
Stamford, CT 06905  
(203) 329-1160

  
David P. Gordon  
Reg. No. 29,995

VERIFIED STATEMENT CLAIMING SMALL ENTITY  
STATUS (37 CFR 1.9(f) and 1.27(b)) - **INDEPENDENT INVENTOR**

As a below named inventor, I hereby declare that I am an independent inventor as defined in 37 CFR 1.9(c) for purposes of paying reduced fees under section 41 (a) and (b) of Title 35, United States Code, to the Patent and Trademark Office with regard to the invention **entitled:**

**REAL TIME DEBUGGER INTERFACE FOR EMBEDDED SYSTEMS**

described in ☒ the specification filed herewith  
☐ application serial No. , filed  
☐ patent No. , issued

I have not assigned, granted, conveyed or licensed and am under no obligation under contract or law to assign, grant, convey or license any rights in the invention to any person who could not be classified as an independent inventor under 37 CFR 1.9(c) if that person had made the invention, or to any concern which would not qualify as a small business concern under 37 CFR 1.9(d) or a nonprofit organization under 37 CFR 1.9(e).

Each person, concern or organization to which I have assigned, granted, conveyed, or licensed or am under an obligation under contract or law to assign, grant, convey or license any rights in the invention is listed below:

☐ no such person, concern, or organization  
☒ persons, concerns or organizations listed below\*

\*Note: Separate verified statements are required from each named person, concern or organization having rights to the invention averring to their status as small entities. (37 CFR 1.27)

**NAME: TranSwitch Corporation**


**ADDRESS: 2 Enterprise Drive, Shelton, CT 06484**

☐ Individual ☒ Small Business Concern ☐ Nonprofit Organization

I acknowledge the duty to file, in this application or patent, notification of any change in status resulting in loss of entitlement to small entity status prior to paying, or at the time of paying, the earliest of the issue fee or any maintenance fee due after the date on which status as a small entity is no longer appropriate. (37 CFR 1.28(b))

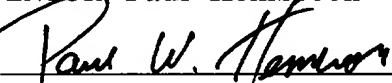
I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application, any patent issuing thereon, or any patent to which this verified statement is directed.

**NAME OF INVENTOR: Subhash C. Roy**

SIGNATURE 

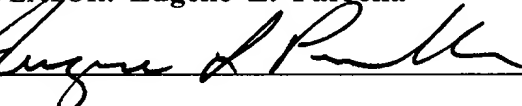
DATE 4/17/98

**NAME OF INVENTOR: Paul Hembrook**

SIGNATURE 

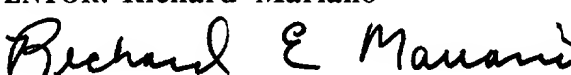
DATE 4/17/98

**NAME OF INVENTOR: Eugene L. Parrella**

SIGNATURE 

DATE 4/17/98

**NAME OF INVENTOR: Richard Mariano**

SIGNATURE 

DATE 4/17/98

**VERIFIED STATEMENT CLAIMING SMALL ENTITY  
STATUS (37 CFR 1.9(f) and 1.27(c) - SMALL BUSINESS CONCERN**

I hereby declare that I am ☐ the owner of the small business concern identified below  
☒ an official of the small business concern empowered to act on behalf of the  
 concern identified below:

**NAME OF CONCERN: TranSwitch Corporation**  
**ADDRESS OF CONCERN: 8 Progress Drive**  
**Shelton, CT 06484**

I hereby declare that the above identified small business concern qualifies as a small business concern as defined in 13 CFR 121.3-18, and reproduced in 37 CFR 1.9 (d), for purposes of paying reduced fees under section 41(a) and (b) of Title 35, United States Code, in that the number of employees of the concern, including those of its affiliates, does not exceed 500 persons. For purposes of this statement, (1) the number of employees of the business concern is the average over the previous fiscal year of the concern of the persons employed on a full-time, part-time or temporary basis during each of the pay periods of the fiscal year, and (2) concerns are affiliates of each other when either, directly or indirectly, one concern controls or has the power to control the other, or a third party or parties controls or has the power to control both.

I hereby declare that rights under contract or law have been conveyed to and remain with the small business concern identified above with regard to the invention, **entitled: REAL TIME DEBUGGER INTERFACE FOR EMBEDDED SYSTEMS**

by inventor(s): **Subhash C. Roy, Paul Hembrook, Eugene L. Parrella and Richard Mariano**  
 described in ☒ the specification filed herewith  
☐ application serial No. , filed  
☐ patent No. , issued

If the rights held by the above identified small business concern are not exclusive, each individual, concern or organization having rights to the invention is listed below\* and no rights to the invention are held by any person, other than the inventor, who could not qualify as a small business concern under 37 CFR 1.9(d) or by any concern which would not qualify as a small business concern under 37 CFR 1.9(d) or a nonprofit organization under 37 CFR 1.9(e).

\*Note: Separate verified statements are required from each named person, concern or organization having rights to the invention averring to their status as small entities. (37 CFR 1.27)

**NAME:**

**ADDRESS:**

☐ Individual ☐ Small Business Concern ☐ Nonprofit Organization

I acknowledge the duty to file, in this application or patent, notification of any change in status resulting in loss of entitlement to small entity status prior to paying, or at the time of paying, the earliest of the issue fee or any maintenance fee due after the date on which status as a small entity is no longer appropriate. (37 CFR 1.28(b))

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application, any patent issuing thereon, or any patent to which this verified statement is directed.

**NAME OF PERSON SIGNING: Santanu Das**  
**TITLE IN ORGANIZATION: President**  
**ADDRESS OF PERSON SIGNING: 14 Hunter Ridge Road**  
**Monroe, CT 06468**

**SIGNATURE:** Santanu Das

**DATE:** 4.17.98

## 1 REAL TIME DEBUGGER INTERFACE FOR EMBEDDED SYSTEMS

2  
3 BACKGROUND OF THE INVENTION  
4

## 5 1. Field of the Invention

6 The invention relates to systems and methods for debugging  
7 software in real time. More particularly, the invention relates  
8 to systems and methods for the real time debugging of firmware in  
9 embedded systems, e.g. ASIC chips having one or more processors on  
10 a single chip.

11  
12 2. State of the Art

13 Software debugging may be accomplished in a number of ways,  
14 some of which are not performed in real time. A traditional  
15 debugging technique is to step through program instructions at a  
16 rate much slower than the rate at which the program is designed to  
17 run in real time. By stepping through the program instructions  
18 one-by-one, errors can be observed as they happen and the program  
19 code lines executed immediately prior to the error can be analyzed  
20 to find the cause of the error. This technique is not helpful,  
21 however, if the error in program execution is the result of timing  
22 errors or other types of errors which only occur when the program  
23 is running at real time speed. As used herein, the term "real

1 time" means the rate at which a program must execute in order to  
2 process the incoming data rate which may be quite high.

3  
4 A widely used technique for debugging a program which is  
5 running in real time is called "tracing". Tracing involves  
6 recording the transactions performed by the computer as it  
7 executes the program code. The trace of activities performed by  
8 the computer during the time of a failure can be a useful guide in  
9 isolating possible causes of the failure.

10  
11 Another useful debugging tool is to set breakpoints at  
12 selected places in the program. The breakpoints trap the flow of  
13 the software and provide insight into whether, when, and how  
14 certain portions of the software are entered and exited. An  
15 analysis of the flow of the software can provide information which  
16 is useful in isolating bugs.

17  
18 Many state-of-the-art tracing and trapping methods are  
19 accomplished by a debug support circuit which is connected to the  
20 system bus, i.e. the bus which couples the CPU to memory. See,  
21 for example, U.S. Patent Number 5,491,793 to Somasundaram et al.  
22 entitled "Debug Support in a Processor Chip." Connecting a debug  
23 circuit to the system bus is convenient because addresses,  
24 instructions, and data can be accessed via the system bus.

1 However, coupling the debug support circuit to the system bus  
2 increases the electrical load on the bus and interferes with the  
3 operation of the bus. Moreover, operation of the system bus may  
4 interfere with operation of the debug support circuit. In  
5 addition, the system bus may not provide all the information  
6 necessary for debugging a program running on a CPU which uses  
7 internal cache. These CPUs will not access the system bus if the  
8 information they need is available in cache. If an error occurs  
9 while the CPU is accessing internal cache, the debug support  
10 circuit will not be able to access the information it needs.

11  
12 Another tracing and trapping method is disclosed in U.S.  
13 Patent Number 5,833,310 to Whistel et al. entitled "On-Chip In-  
14 Circuit-Emulator Memory Mapping and Breakpoint Register Modules."  
15 According to this method, an internal bus controller is coupled to  
16 the memory address bus and a match register. When a memory  
17 address written to the address bus matches an address in the match  
18 register, a memory mapping module maps a memory cycle to an  
19 external debug memory. The user can set specific bus event  
20 conditions for which memory is mapped by writing to a set of  
21 breakpoint registers. A disadvantage of this method is that it  
22 requires an additional set of I/O pins for the chip so that the  
23 external debug memory can be coupled to the chip. This may

1 require a significant number of pins since the addresses to be  
2 mapped may be 32 or 64 bits wide.  
3

4 Still another tracing and trapping method is disclosed in  
5 U.S. Patent Number 5,513,346 to Satagopan et al. entitled "Error  
6 Condition Detector for Handling Interrupt in Integrated Circuits  
7 Having Multiple Processors." According to this method, an  
8 interrupt processor controller intercepts all interrupts and  
9 routes them to the appropriate processor in a multiprocessor chip.  
10 The interrupt processor controller includes logic which determines  
11 when an interrupt will cause an error because a previously  
12 instigated interrupt has not been cleared. When such an error is  
13 detected, a bit is set in an error detect register, the bit  
14 corresponding to an interprocessor interrupt channel. The bits in  
15 the register are ORed and a single bit output indicates the  
16 occurrence of an error. The register may then be examined to  
17 determine the location of the interrupt error in the executing  
18 code. This method does not interfere with the system bus and does  
19 not require very many additional pins on the chip. However, the  
20 debugging information that it provides is limited.  
21

22 The Motorola MPC-860 PowerQuicc™ includes a program  
23 development system interface port which provides a three bit  
24 output indicative of the state of the program execution as the

1 program is being executed. The MPC-860 is a 40 mHz communications  
2 controller but the development system interface port is only  
3 operable at a rate of 4 mHz. Thus, the port can not be used for  
4 real time debugging. The specifications for the MPC-860 are found  
5 in the "MPC-860 POWERQUICC USER'S MANUAL", Copyright 1996  
6 Motorola, Inc., Schaumburg, IL, the complete disclosure of which  
7 is incorporated herein by reference.

8  
9 ASIC design using one or more embedded processors poses  
10 additional debugging challenges. The prior art methods of  
11 trapping instructions at a given point in time implies that the  
12 system must be stopped to allow debugging of firmware. Once the  
13 system is stopped, however, real time events and their timing  
14 relationships are lost. If there is a firmware bug which is only  
15 identifiable in the presence of live traffic (during real time  
16 operations) it is necessary to obtain contextual information about  
17 the error before the firmware is changed.

#### 18 19 SUMMARY OF THE INVENTION 20

21 It is therefore an object of the invention to provide a  
22 debugging interface for tracing instructions without loss of real  
23 time context and event interaction.  
24



1       It is also an object of the invention to provide a debugging  
2 interface which does not interfere with the operation of a  
3 processor or system bus.

4  
5       It is another object of the invention to provide a debugging  
6 interface which does not require many additional pins on a  
7 processor chip.

8  
9       It is a further object of the invention to provide a  
10 debugging interface which provides access to a substantial amount  
11 of information about the executed instructions.

12  
13       In accord with these objects which will be discussed in  
14 detail below, the debugging interface of the present invention  
15 includes a first decoder coupled to the sequencer of a processor  
16 and to the Instruction RAM (IRAM) of the processor. The first  
17 decoder, according to the invention, provides a real time three  
18 bit output on a cycle by cycle basis which is indicative of the  
19 processor activity during the last clock cycle. According to a  
20 presently preferred embodiment, the three bit output indicates  
21 seven different conditions regarding processor activity. In  
22 particular, the three bit output indicates whether or not a new  
23 instruction has been executed since the last clock cycle, and if a  
24 new instruction has been executed, whether the last instruction

1   executed by the processor was an immediate jump, a jump to  
2   register, or a branch taken. In addition, the three bit output  
3   will indicate whether execution of the instruction resulted in an  
4   exception. By recording this three bit output over time, and  
5   comparing it to the actual instructions listed in the program  
6   code, important debugging information is obtained about a program  
7   which was running in real time.

8  
9       According to a preferred embodiment of the invention, a  
10   second decoder and an event history buffer are coupled to the  
11   cause register of the sequencer of the processor. In particular,  
12   the second decoder is coupled to the enable input of the history  
13   buffer and the cause register is coupled to the data input of the  
14   history buffer. The second decoder decodes the contents of the  
15   cause register and enables the history buffer whenever the  
16   contents of the cause register indicates an exception, a jump  
17   register instruction, or a change in the status of an interrupt  
18   line. Whenever the history buffer is enabled, information from  
19   the cause register and the program counter is loaded into the  
20   buffer. By recording the contents of the history buffer over  
21   time, and comparing the information to the actual program code,  
22   additional important debugging information is obtained about a  
23   program which was running in real time. According to this  
24   preferred embodiment of the invention, the seventh condition

1 indicated by the three bit output of the first decoder is whether  
2 an exception was encountered without writing to the history  
3 buffer.

4  
5 According to the presently preferred embodiment, each entry  
6 in the event history buffer is forty-four bits. Each forty-four  
7 bit entry in the history buffer includes the current sixteen bit  
8 time stamp, twenty three bits from certain fields of the cause  
9 register or program counter, one bit indicating whether the entry  
10 is related to a jump or an exception, two bits identifying the  
11 processor number (in a multiprocessor system), one bit identifying  
12 whether the history buffer has overflowed, and a time stamp  
13 rollover bit. The history buffer preferably has a depth of at  
14 least sixteen entries.

15  
16 An exemplary implementation of the debugging interface is  
17 embodied on an ASIC chip having three processors. Each processor  
18 is provided with two decoders as described above and a single  
19 event history buffer is provided on the chip. Nine pins on the  
20 chip are used to provide access to the three bit outputs of each  
21 first decoder. Three pins on the chip provide serial access  
22 (data, clock, and enable) to the contents of the event history  
23 buffer. These twelve pins on the chip allow a diagnostic device  
24 to be coupled to the chip during real time operations without

1 interfering with the operation of the chip. The outputs of the  
2 first decoders and the contents of the event history buffer can be  
3 recorded over time by the diagnostic device to provide a real time  
4 record of the processing events occurring in the chip during real  
5 time. This real time record taken together with knowledge of the  
6 program code being executed provides a true picture of the  
7 processors' execution sequence in real time and thereby expedite  
8 debugging of code.

9  
10 Additional objects and advantages of the invention will  
11 become apparent to those skilled in the art upon reference to the  
12 detailed description taken in conjunction with the provided  
13 figures.

#### 14 15 BRIEF DESCRIPTION OF THE DRAWINGS

16  
17 Figure 1 is a schematic block diagram of an exemplary  
18 implementation of a real time debugger interface according to the  
19 invention; and

20  
21 Figure 2 is a schematic block diagram of a debugging system  
22 coupled to a chip embodying a real time debugger interface  
23 according to the invention.

## 1 DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

2  
3 Referring now to Figures 1, an exemplary ASIC chip 10  
4 incorporating a debugger interface according to the invention  
5 includes three processors 12a, 12b, 12c, sharing a common clock 16  
6 via a clock bus 17. Each processor includes an instruction RAM  
7 (IRAM) 18a, 18b, 18c, an arithmetic logic unit (ALU) 20a, 20b,  
8 20c, and a "sequencer" 22a, 22b, 22c. Each sequencer includes a  
9 program counter 24a, 24b, 24c and a cause register 26a, 26b, 26c.  
10 Each program counter contains an index of the instructions in an  
11 associated IRAM and a pointer to the index as the instructions are  
12 executed by the processor. The cause registers store current  
13 information about interrupts, exceptions, and other processor  
14 functions.

15  
16 According to one aspect of the invention, a first decoder  
17 28a, 28b, 28c is coupled to each IRAM 18a, 18b, 18c, and to each  
18 sequencer 22a, 22b, 22c, i.e., to each program counter and each  
19 cause register. Each first decoder has a three bit output 30a,  
20 30b, 30c which is available off the chip 10 via three pins (0, 1,  
21 2) in real time.

22  
23 As mentioned above, the three bit output of each first  
24 decoder 28 provides an indication of the processor activity during

1 the last clock cycle. Thus, the decoder 28 is arranged to  
2 indicate whether the program counter has moved its pointer to a  
3 new instruction. The decoder also decodes the instruction in the  
4 IRAM to provide information about the instruction, and decodes the  
5 contents of the cause register to provide an indication of an  
6 exception encountered during the execution of an instruction.  
7 According to a presently preferred embodiment, the first decoder  
8 28 generates a three bit output which is interpreted as shown in  
9 Table 1, below.

Output	Mnemonic	Description
000	NC	No Change
001	INC	Program Counter Increment
010	JI	Program Counter Jump Immediate
011	JR	Program Counter Jump Register
100	ECP	Exception Encountered
101	PBT	Program Counter Branch Taken
110	RSD	Reserved
111	ENH	Exception Encountered, No History Buffer Entry Written

Table 1

The output 000 indicates that there has been no change in the processor since the last clock cycle; i.e., the processor has not processed a new instruction and the program counter pointer has not changed. The output 001 indicates that the processor has processed the next instruction in the program; i.e., the program counter pointer has incremented to the next instruction in the index. The output 010 indicates that the last instruction processed by the processor was a "hard coded" jump to an instruction; i.e., the instruction in IRAM pointed to by the

1 program counter includes code indicating that it is a jump  
2 instruction to an absolute address in the program. The output 011  
3 indicates that the last instruction processed by the processor was  
4 a jump to an instruction based on the contents of a register;  
5 i.e., the instruction in IRAM pointed to by the program counter  
6 includes code indicating that it is a jump instruction to a  
7 location in the program determined by the value of a variable.  
8 The output 100 indicates that since the last clock cycle the  
9 processor has encountered an interrupt or an exception; i.e., the  
10 contents of the cause register contain code which indicates an  
11 interrupt or exception. The output 101 indicates that the last  
12 instruction processed by the processor was a pc branch taken;  
13 i.e., the instruction in IRAM pointed to by the program counter  
14 includes code indicating that it is a branch back to another  
15 instruction. The output 110 is not presently used, but is  
16 reserved for future use. The output 111 indicates that since the  
17 last clock cycle the processor has encountered an interrupt or an  
18 exception; and that no entry was made in the history buffer

19  
20 The operation of the first decoder 28 and its output is  
21 illustrated with reference to a simple code listing which is shown  
22 below in Table 2.

23



LINE NUMBER	INSTRUCTION
10	Input A
20	B=5
30	C=2
40	D=B+C
50	If D=7 then Goto 70
60	Goto A*10
70	B=4
80	Goto 30
90	End

Table 2

The listing in Table 2 has one "immediate" or "hard coded" jump instruction at line 80 and a conditional branch at line 50. It also has one jump instruction, line 60, based on the contents of a register, i.e. the value of A which is input at line 10. The three bit output of the first decoder during execution of the instructions shown in Table 2 is illustrated in Table 3 below where the values of variables A, B, C, and D are also shown.

Current Line	Next Line	A	B	C	D	Mnemonic	Three Bit Output
10	20	?	?	?	?	INC	001
20	30	?	5	?	?	INC	001
30	40	?	5	2	?	INC	001
40	50	?	5	2	7	INC	001
50	70	?	5	2	7	PBT	101
70	80	?	4	2	7	INC	101
80	30	?	4	2	7	JI	010
30	40	?	4	2	7	INC	001
40	50	?	4	2	6	INC	001
50	60	?	4	2	6	INC	001
60	?	?	4	2	6	JR	011

Table 3

When the first instruction (listed in line 10) is executed, the first decoder indicates that a program counter increment (INC) in the execution of the program has occurred and shows an output of "001". As the program progresses from the instruction on line 10 through the instruction on line 40, the first decoder continues to indicate that a program counter increment (INC) in the execution of the program has occurred and continues to show an output of "001". When the instruction on line 50 is executed, the first decoder indicates that a program counter branch taken (PBT)

1 has occurred and shows an output of "101". As seen in Tables 2  
2 and 3, the program branches to line 70 because the conditional  
3 expression of line 50 is true based on the variable D=7. Upon  
4 execution of line 70, the first decoder indicates that a program  
5 counter increment (INC) in the execution of the program has  
6 occurred and shows an output of "001". When the instruction on  
7 line 80 is executed, the first decoder indicates that an immediate  
8 jump (JI) has occurred and shows an output of "010". As seen in  
9 Tables 2 and 3, the program jumps to line 30. When the  
10 instructions on lines 30 and 40 are executed, the first decoder  
11 indicates that a program counter increment (INC) in the execution  
12 of the program has occurred and shows an output of "001". When  
13 line 50 is executed (now for the second time) the first decoder  
14 indicates that a program counter increment (INC) in the execution  
15 of the program has occurred and shows an output of "001" because  
16 the condition (D=7) for the jump in line 50 is no longer valid.  
17 Line 60 is now executed and a jump to a location stored in a  
18 register occurs. The first decoder therefore indicates a jump to  
19 register (JR) by showing an output of "011".

20

21 Referring once again to Figure 1, according to another aspect  
22 of the invention, each cause register 26a, 26b, 26c is coupled to  
23 the data input D of an event history buffer 14 and a second  
24 decoder 32a, 32b, 32c is coupled to each cause register and to the

1 enable input E of the history buffer 14. The clock 16 provides  
2 the common clock signal to the clock input C of the history buffer  
3 14 via the clock bus 17, and a timestamp register 19 is also  
4 coupled to the clock bus 17. The contents of the history buffer  
5 14 are made available off chip by three pins for the data, clock,  
6 and enable (D, C, E) of the history buffer 14. According to this  
7 aspect of the invention, when certain conditions are detected by  
8 one of the second decoders 32, the history buffer is enabled via  
9 the appropriate decoder, and information from the cause register,  
10 the timestamp register, and the program counter is stored in the  
11 history buffer. More particularly, the second decoder 32 enables  
12 the history buffer whenever the first decoder contains code which  
13 indicates that the processor is processing an instruction to jump  
14 to a location stored in a register, whenever the first decoder  
15 contains code indicating an exception was encountered, and  
16 whenever the first decoder contains code indicating a change in  
17 state of an interrupt line.

18  
19 According to a presently preferred embodiment, when the  
20 history buffer is enabled, it captures forty-four bits of  
21 information from the cause register or program counter, and the  
22 timestamp register. The forty-four bits of information are  
23 preferably organized as illustrated in Table 4 below.

43	42	41	40 - 18	17	16	15 - 0
Mode	Proc	Cause/PC	HOVRF	TR	Time Stamp	

Table 4

The first bit, bit location 43, is a mode identifier indicating whether the entry being stored has program counter information or cause register information. A two bit processor identification number is stored in binary form at bit locations 42, 41. This number is used to indicate which processor's information is being stored (in the case of a multiprocessor system). The next twenty-three bits at bit locations 40 through 18 are used to store cause register information or program counter information depending on the mode as explained above. If program counter information is being stored, the contents of the program counter are stored at bit locations 40 through 18. If cause register information is being stored, bit location 40 is used to indicate whether the exception occurred while the processor was executing an instruction in the branch delay slot. (This applies to pipelined processors such as RISC processors.) Bit locations 39 through 35 are used to store processor related exception conditions. Bit locations 34 through 18 are used to store an indication of all pending interrupts (external, software, co-processor. The HOVRF field at bit location 17 is used to indicate

1 whether the internal event history buffer has overflowed. The TR  
2 bit 16 is used to indicate a timestamp rollover and bits 15  
3 through 0 are used to store a sixteen bit timestamp. According to  
4 the presently preferred embodiment, the forty-four bits captured  
5 in the history buffer 14 are serially output on data pin D over  
6 forty-four clock cycles (bit serial output).

7  
8 As mentioned above, the event history buffer records  
9 information when an event (either an unmasked exception or a PC  
10 jump register instruction) has occurred. According to a presently  
11 preferred embodiment, this requires an additional mask register  
12 per cause register and a free running timestamp counter. The  
13 event masks are provided by a JTAG test register load instruction  
14 in the static debug interface. When the cause register bits  
15 corresponding to an exception are unmasked or a PC jump register  
16 instruction is encountered, an entry is made in the history  
17 buffer.

18  
19 Those skilled in the art will appreciate that the outputs of  
20 the first decoder 28 and the contents of the history buffer 14  
21 provide a relatively complete indication of each processor's  
22 execution sequence in real time, particularly when viewed in light  
23 of the actual program code which is being executed. Therefore,

1 according to the invention, a debugging system may be coupled to  
2 the first decoders and history buffer as illustrated in Figure 2.

3  
4 Turning now to Figure 2, the outputs 30a, 30b, 30c of the  
5 first decoders and the D,C,E terminals of the history buffer are  
6 coupled to a debugging computer 44 which preferably has a copy of  
7 the program code stored therein. The three-bit outputs 30a,30b,  
8 30c of the first decoders and the D,C,E terminals of the history  
9 buffer are preferably coupled to an interface buffer 40 which is  
10 coupled by a serial, parallel, or network connection 42 to the  
11 debugging computer 44. The interface buffer 40 is a rate  
12 decoupling buffer. In a present embodiment of the invention, the  
13 debugger interface is provided on a 100 MHz three processor  
14 system. In that system, the data rate for reading the event  
15 history buffer is approximately 1 gigabit/sec. Current PCs cannot  
16 keep up with that data rate. Therefore, the buffer 40 is provided  
17 to prevent the loss of event history data.

18  
19 As the program is running on the ASIC 10, the debugging  
20 computer 44 collects information from the first decoders and the  
21 history buffer. The information collected by the computer 44 is  
22 associated with each line of code being executed by the ASIC by  
23 stepping through the copy of the code which is stored in the  
24 computer 44. When a bug is encountered, the complete history of

1 instruction execution leading up to the failure can be reviewed  
2 with the computer 44. The debugging system is non-invasive and  
3 permits debugging of programs operating in real time.  
4

5 There have been described and illustrated herein embodiments  
6 of a real time debugger interface for embedded systems. While  
7 particular embodiments of the invention have been described, it is  
8 not intended that the invention be limited thereto, as it is  
9 intended that the invention be as broad in scope as the art will  
10 allow and that the specification be read likewise. Thus, while  
11 particular encoding schemes have been disclosed with reference to  
12 the first decoder output and the history buffer contents, it will  
13 be appreciated that other encoding schemes could be utilized  
14 provided that they achieve substantially the same results as  
15 described herein. Also, while the invention has been illustrated  
16 with reference to a three-processor ASIC chip, it will be  
17 recognized that the invention may be applied in other types of  
18 chips having greater or fewer processors. Moreover, while  
19 particular configurations have been disclosed in reference to the  
20 indications provided by the first decoders, it will be appreciated  
21 that other configurations could be used as well, provided that  
22 they achieve substantially the same results as described herein.  
23 It will therefore be appreciated by those skilled in the art that



- 1 yet other modifications could be made to the provided invention
- 2 without deviating from its spirit and scope as so claimed.

## Claims:

1. A processor having a real time debugging interface, said processor comprising:

a) instruction memory means for storing instructions to be executed by said processor;

b) program counter means coupled to said instruction memory means for indexing said instructions;

c) cause register means for indicating information regarding interrupts and exceptions; and

d) first decoder means for indicating information about an instruction executed by said processor during a clock cycle, said first decoder means being coupled to said instruction memory means, said program counter means, and said cause register means, said first decoder means having a first output, wherein

said first output provides information regarding activity of said processor in real time.

2. A processor according to claim 1,

said information regarding processor activity includes information as to at least one of a jump instruction has been executed, a jump instruction based on the contents of a register has been executed, a branch has been taken, and an exception has been encountered.

3. A processor according to claim 1, wherein:  
     said clock cycle is a processor clock cycle, and  
     said first decoder means updates said information about each instruction executed by said processor for each said processor clock cycle.
  
4. A processor according to claim 3, wherein:  
     said information about each instruction executed by said processor includes an indication whether or not an instruction has been executed since a previous processor cycle.
  
5. A processor according to claim 1, wherein:  
     said first output is a three bit parallel output.

6. A processor according to claim 1, further comprising:

e) second decoder means coupled to said cause register means for indicating information about contents of said cause register means, said second decoder means having a second output; and

f) event history buffer means for storing information regarding processor events, said event history buffer means having a data input, a data output, and an enable input, said data input being coupled to said cause register means and said enable input being coupled to said second output, wherein

said second decoder means decodes contents of said cause register means and enables said event history buffer means to capture contents of said cause register means when contents of said cause register means indicate a particular event.

7. A processor according to claim 6, wherein:

said second decoder means enables said event history buffer means when contents of said cause register means indicate an event including at least one of a change in status of an interrupt line, an internal processor exception, and a jump instruction based on the contents of a register.

8. A processor according to claim 6, wherein:

said data output of said event history buffer means is a bit serial output.

9. A processor according to claim 6, wherein:

said processor is embodied on a chip having a plurality of pins,

said first output and said data output are provided via some of said plurality of pins.

10. A processor according to claim 9, wherein:

said first output is an n-bit parallel output, and  
said data output is a serial output.

11. An embedded system having a plurality of processors and a real time debugging interface, said system comprising:

a) a plurality of instruction memory means for storing instructions to be executed by a respective one of said plurality of processors;

b) a plurality of program counter means, each coupled to a respective one of said plurality of instruction memory means for indexing contents of said instruction memory means;

c) a plurality of cause register means for indicating information regarding interrupts and exceptions for a corresponding one of said plurality of processors, each of said cause register means being coupled to a respective one of said processors; and

d) a plurality of first decoder means, each said first decoder means coupled to a respective one of said instruction memory means, to a respective one of said program counter means, and a respective one of said cause register means, each said first decoder means for indicating information about an instruction executed during a clock cycle by a respective one of said processors, each said first decoder means having a first output, wherein

each said first output provides information regarding activity of said processor in real time.

12. An embedded system according to claim 11, wherein:

said information regarding processor activity includes information as to at least one of a jump instruction has been executed, a jump instruction based on the contents of a register has been executed, a branch has been taken, and an exception has been encountered.

13. An embedded system according to claim 11, wherein:

said clock cycle is a processor clock cycle, and

each said first decoder means updates said information about each instruction executed by a respective processor for each said processor clock cycle of said respective processor.

14. An embedded system according to claim 13, wherein:

each said information about each instruction executed by a respective processor includes an indication whether or not an instruction has been executed since a previous processor cycle of said respective processor.

15. An embedded system according to claim 11, wherein:

each of said first outputs is a three bit parallel output.

16. An embedded system according to claim 11, further comprising:

e) a plurality of second decoder means, each coupled to a respective one of said plurality of cause register means, each said second decoder means for indicating information about contents of a respective cause register means; and

f) an event history buffer means for storing information regarding processor events, said history buffer means having a data input, a data output, and an enable input, said data input being coupled to each of said plurality of cause register means and said enable input being coupled to each of said second outputs, wherein

each of said second decoder means decodes contents of a respective cause register means and enables said event history buffer to capture contents of said respective cause register means when contents of said respective cause register means indicate a particular event.

17. An embedded system according to claim 16, wherein:

each said second decoder means enables said event history buffer means when contents of a respective cause register means indicate an event including at least one of a change in status of an interrupt line, an internal processor exception, and a jump instruction based on the contents of a register.



18. An embedded system according to claim 16, wherein:  
     said data output of said event history buffer means is a bit serial output.
19. An embedded system according to claim 11, wherein:  
     said system is embodied on a chip having a plurality of pins,  
     said first and second outputs are provided via some of said plurality of pins.
20. An embedded system according to claim 19, wherein:  
     each of said first outputs is an n-bit parallel output, and  
     said second output is a serial output.
21. A method of debugging a processor, said method comprising:
  - a) providing information about processor activity in real time;  
 and
  - b) associating the instructions executed by the processor with the information about processor activity.

22. A method according to claim 21, wherein:

said step of providing information about processor activity includes providing information about every instruction executed by the processor.

23. A method according to claim 22, wherein:

said step of providing information about processor activity includes providing information that the processor has not executed an instruction during the last processor cycle.

24. A method according to claim 21, wherein:

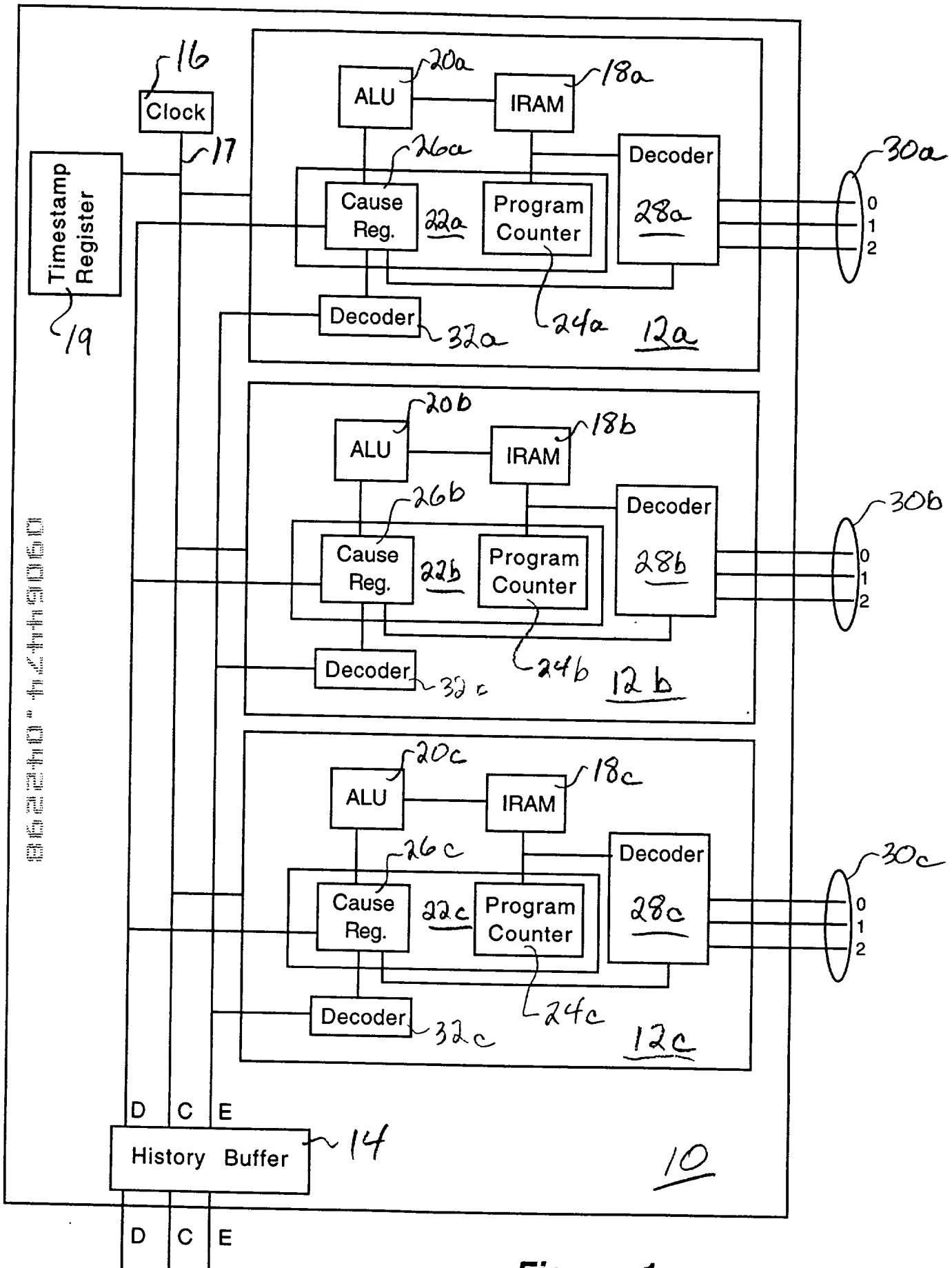
the information about processor activity includes an indication of at least one of whether the last instruction executed was a jump, a jump based on the contents of a register, a branch taken, or an instruction which encountered an exception.

25. A method according to claim 21, further comprising:

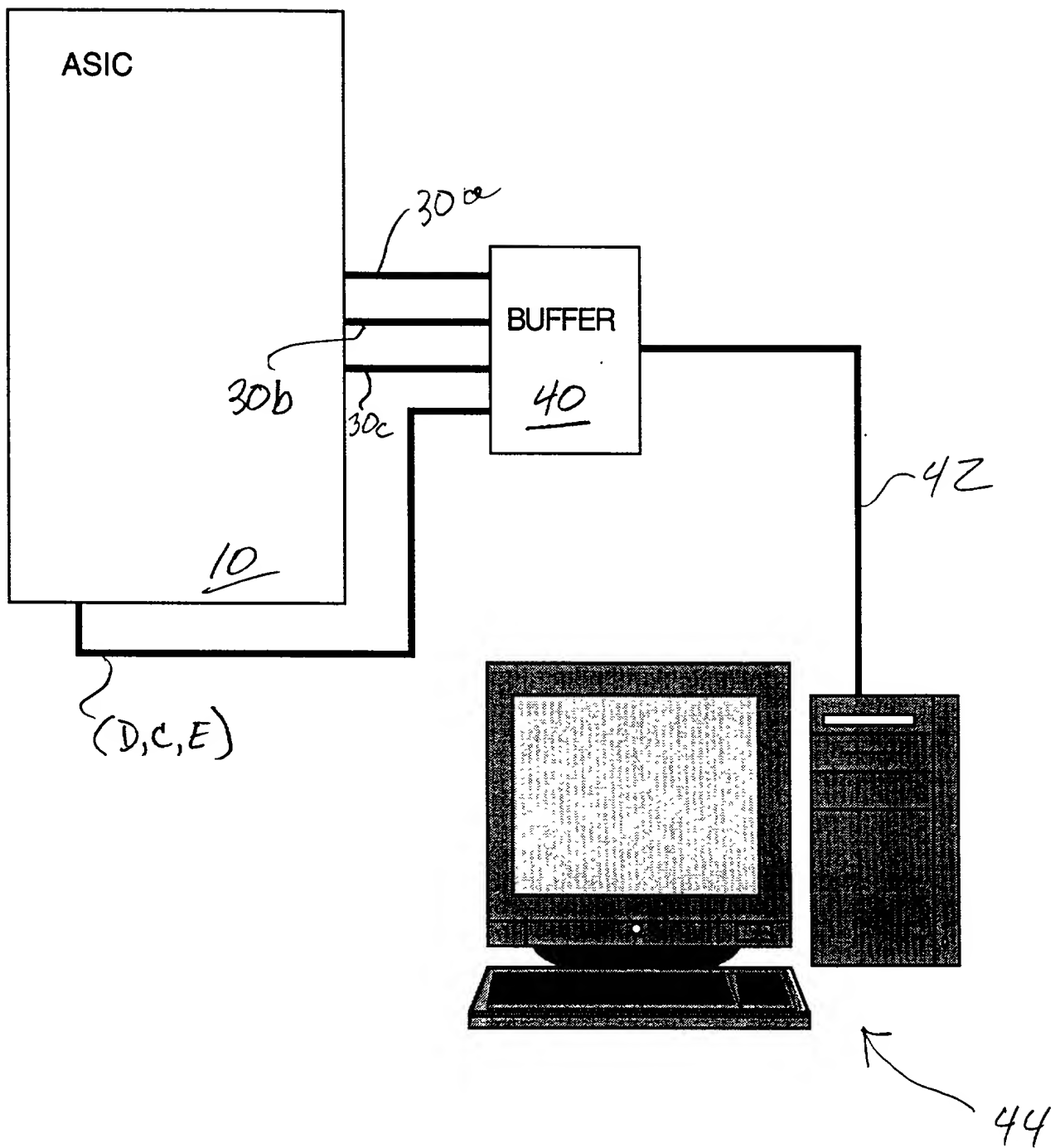
c) providing information regarding the status of the processor when certain processor events occur, said certain processor events including at least one of a change in status of an interrupt line, an internal processor exception, and the execution of a jump instruction based on the contents of a register.

## 1 ABSTRACT OF THE DISCLOSURE

2  
3 A debugging interface includes a pair of decoders and an  
4 event history buffer coupled to the sequencer of a processor. The  
5 first decoder is coupled to the program counter of the sequencer  
6 and the Instruction RAM of the processor. The second decoder is  
7 coupled to the cause register of the sequencer and the event  
8 history buffer is also coupled to the cause register. The first  
9 decoder provides a three bit real time output which is indicative  
10 of the processor activity on a cycle by cycle basis. The three  
11 bit output indicates seven different conditions: whether the last  
12 instruction executed by the processor was an inc, an exception, an  
13 exception with no event history buffer entry, or a branch taken,  
14 whether there has been no instruction executed since the last  
15 clock cycle, and whether a jump was an immediate jump or a jump to  
16 a register. The event history buffer is loaded with more detailed  
17 information about the instruction last executed when the first  
18 decoder indicates that the last instruction was an exception or a  
19 jump to a register, and when there is a change in state of an  
20 interrupt line or an internal processor exception. An exemplary  
21 implementation of the debugging interface is embodied on an ASIC  
22 chip having three processors. Each processor is provided with a  
23 first and second decoders and a single event history buffer for  
24 all processors is provided on the chip.



**Figure 1**



**Figure 2**

**DECLARATION FOR PATENT APPLICATION AND POWER OF ATTORNEY**

As below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below next to my name, and

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed for and for which a patent is sought on the invention entitled

**REAL TIME DEBUGGER INTERFACE FOR EMBEDDED SYSTEMS,**

the specification of which

☒ is attached hereto.

☐ was filed on \_\_\_\_\_

as application Serial Number \_\_\_\_\_

and was amended on (if applicable) \_\_\_\_\_

I hereby state that I have reviewed and understand the contents of the above identified specification, including the claims, as amended by an amendment referred to above.

I acknowledge the duty to disclose information which is material to the examination of this application in accordance with Title 37, Code of Federal Regulations, Section 1.56(a).

I verify that I am qualified as an independent inventor under Title 37, Code of Federal Regulations, Section 1.9(c), and my obligation to assign rights to this invention, if any, is to a qualified small business concern under Title 37, Code of Federal Regulations, Section 1.9(d).

I hereby claim foreign priority benefits under Title 35, United States Code, Section 119 of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

**Prior Foreign Application(s)**

**Priority Claimed**

_____ (Number)	_____ (Country)	____/____/____ D/M/YR FILED	<input type="checkbox"/> YES	<input type="checkbox"/> NO
-------------------	--------------------	--------------------------------	------------------------------	-----------------------------

_____ (Number)	_____ (Country)	____/____/____ D/M/YR FILED	<input type="checkbox"/> YES	<input type="checkbox"/> NO
-------------------	--------------------	--------------------------------	------------------------------	-----------------------------

I hereby claim the benefit under Title 35, United States Code, Section 120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, Section 112, I

062340-4449060



**THIRD JOINT INVENTOR**

Signature

*Eugene L. Parrella*

Date

*4/17/98*

Full Name Eugene L. Parrella

Residence 48 Settlers Farm Road, Monroe, CT 06468

Citizenship US

P.O. Address same as residence

**FOURTH JOINT INVENTOR**

Signature

*Richard E. Mariano*

Date

*4/17/98*

Full Name Richard Mariano

Residence 140 Codfish Hill Road, Bethel, CT 06801

Citizenship US

P.O. Address same as residence

062340" 42449060